เรื่องน่าอ่าน
ดร.ปัญญา เปรมปรีดิ์

ภาษาโปรแกรม นั้นสำคัญนักหรือ

ทำความเข้าใจกับการใช้ภาษาสำหรับการเขียนโปรแกรมความสำคัญอยู่ที่ไหน

ท่านที่ไม่ใช่นักคอมพิวเตอร์ และท่านที่เพิ่งจะเริ่มเข้ามาทำงานด้านโปรแกรมมิงมักจะได้รับฟังนักคอมพิวเตอร์เขาเถียงกันถึงข้อดีข้อเสียของภาษาต่าง ๆ โดยเฉพาะอย่างยิ่ง พวกที่กำลังจะเอาไมโครคอมพิวเตอร์และไฮเพาเวอร์เวอร์กสเตชันเข้ามาใช้งาน พวกหนึ่งบอกว่าต้องใช้ภาษาซี อีกพวกหนึ่งบอกว่าไม่จำเป็น ท่านที่ยังไม่เคยจับงานด้านเขียนโปรแกรมอย่างแท้จริงก็เลยงง ๆ แล้วก็มีหลายคนหันมาถามผมว่า ผมคิดอย่างไร และมันจะมีผลกระทบต่องาน และต่อคนที่จะไหลเข้ามาอยู่ในวิชาชีพนักเขียนโปรแกรมมากน้อยแค่ไหน เรื่องนี้ผมคิดว่าเราคงต้องมานั่งทำความเข้าใจกันพอสมควร ทั้งนี้เพราะเป็นเรื่องที่เข้าใจยาก และมีสถานการณ์ที่ต้องพิจารณาหลายรูปแบบ

รู้จักภาษาสำหรับเขียนโปรแกรม

เรามาเริ่มจากการสำรวจความคิดเห็นอย่างคร่าว ๆ กันก่อนดีไหม? ภาษาที่เราใช้เขียนโปรแกรมในปัจจุบันมีอยู่ 6-7 ภาษา คือ แอสเซมเบลอร์ อาร์พีจี ฟอร์แทรน โคบอล เบสิก ปาสคาล ซี แต่ถ้าเราจะมองให้กว้างออกไปอีกหน่อยก็คงต้องนับเอาภาษาดีเบส โลตัส และเควอรี่แลงเกวจเข้ามาด้วย ผมจะไม่ขอพูดถึงภาษาที่ตายหรือกึ่งตายต่าง ๆ เช่น เอพีแอล พีแอลวัน อัลกอร์ แมด ฌูเวียล และอะไรต่อมิอะไรอีกเยอะแยะ ทั้งนี้เพราะภาษาที่เขาเลิกนิยมแล้วนี้ไม่มีประโยชน์สำหรับคนในปัจจุบัน สิ่งเหล่านี้เป็นประวัติศาสตร์ไปแล้ว

แอสเซมเบลอร์

ภาษาแอสเซมเบลอร์นั้นเขาว่ากันว่าจะดีมากในด้านประสิทธิภาพ ทั้งนี้เพราะเราเข้าถึงคำสั่งในระดับภาษาเครื่องได้ แต่คนส่วนใหญ่ก็ยอมรับว่าเขียนยาก เพราะเขียนผิดได้ง่าย ปัจจุบันนี้ก็เริ่มจะหมดไปจากความนิยมแล้ว จะเหลือคนเพียงไม่กี่คนที่ยังยืนยันว่าดีและจะยังขอใช้อยู่ต่อไป

อาร์พีจี

ภาษาอาร์พีจีนั้นเขาว่าเขียนง่าย มีลอจิกที่วนอยู่ในลูปตายตัว แต่เหลือใช้อยู่ในเครื่องขนาดเล็กเท่านั้น แต่ทุกคนยอมรับว่าเป็นภาษาที่ช้าและอาจเข้าใจยากสำหรับคนบางคน

ฟอร์แทรน

ภาษาฟอร์นแทรนถูกจัดให้เป็นภาษาที่ทำการคำนวณได้ดี แต่มีที่ใช้ไม่มากนัก มีเหลืออยู่แต่ในเครื่องเมนเฟรมกับเครื่องขนาดเล็กของไอบีเอ็มเท่านั้น ภาษานี้คนส่วนใหญ่ยอมรับว่าเขียนผิดได้ง่าย ทำให้ต้องใช้เวลาดีบักนาน

โคบอล

ภาษาโคบอล ถูกจัดให้เป็นภาษาทางด้านธุรกิจทั่วไป ในเมนเฟรมและมินิคอมพิวเตอร์เกือบทุกยี่ห้อจะมีใช้ ภาษานี้มักจะถูกขยายให้รับคำสั่งในภาษาฐานข้อมูลได้ด้วย และในเครื่องระดับไมโครคอมพิวเตอร์และไฮเพาเวอร์เวอร์กสเตชันบางยี่ห้อก็มีใช้ แต่หลาย ๆ คนบอกว่าเป็นภาษาที่เยิ่นเย้อ และไม่ค่อยมีประสิทธิภาพ

เบสิก

ภาษาเบสิก นั้นจริง ๆ แล้วมีหลายรุ่น หลายแนวคิด รุ่นแรก ๆ เป็นแบบโต้ตอบ (Interactive) จากเทอร์มินัล แต่รุ่นนั้นหมดความนิยมไปแล้ว ภาษาเบสิกในปัจจุบันจะเป็น GWBASIC หรือ BASICA ซึ่งใช้ในเครื่องระดับไมโครคอมพิวเตอร์เท่านั้น ภาษานี้เขาว่าเข้าใจง่าย เหมาะสำหรับทำโปรแกรมสั้น ๆ ทั้งนี้เพราะเมื่อเขียนเป็นโปรแกรมใหญ่ ๆ แล้วจะมีข้อผิดพลาดบกพร่อง (บัก) ได้ง่าย และมีแล้วหาที่ผิดได้ยาก งานในเครื่องไมโครคอมพิวเตอร์ส่วนใหญ่ยังเป็นภาษานี้ แต่มีแนวโน้มที่ลดลง

ปาสคาล

ปาสคาล เป็นภาษาที่เกิดมานานแล้ว แต่รุ่นแรก ๆ เข้าใจยาก แถมทำงานช้ามาก เขาจึงไม่นิยมกัน แต่เมื่อมีเครื่องไมโครคอมพิวเตอร์ออกมาสู่ตลาด เราก็หันกลับมานิยมใช้ ทั้งนี้เพราะเป็นภาษาที่มีโครงสร้างดี ทำงานได้รวดเร็วเร็วทั้งด้านคอมพายล์และวิ่ง (Execute) แต่ภาษานี้คนจำนวนมากบอกว่าเข้าใจยาก เขียนโปรแกรมผิดแล้วหาที่ผิดได้ยาก

ภาษาซี

ภาษาซี เป็นภาษาใหม่ มีแหล่งกำเนิดพร้อมกับระบบปฏิบัติการยูนิกซ์ คนจำนวนมากบอกว่ามีประสิทธิภาพสูง แถมยังคล่องตัวมากเมื่อใช้อยู่ในเครื่องที่ใช้ระบบยูนิกซ์ แต่ภาษานี้ก็มีใช้ในเครื่องไมโครคอมพิวเตอร์ธรรมดา และมีใช้ในระบบเครือข่ายไมโครคอมพิวเตอร์ (LAN) ด้วย หลาย ๆ คนบอกว่าภาษานี้ดีมาก เพราะสามารถสนับสนุนเทคนิค "อูีปส์ (OOPS)" แต่หลาย ๆ คนก็ยอมรับว่าเขียนยาก เพราะภาษานี้ครอบคลุมภาษาปาสคาลเข้าไปเกือบเต็มตัว ส่วนเกินของภาษาซีจะได้แก่การติดต่อสื่อสารข้ามเครื่อง

ดีเบส

ดีเบสนั้นมองเห็นได้ชัดเจนว่าเป็นเป็นภาษาเพื่อเขียนโปรแกรม แต่มาเพิ่มคำสั่งที่เกี่ยวกับการสร้างไฟล์ และการดึงข้อมูลจากไฟล์มาใช้งาน หลาย ๆ คนบอกว่าเข้าใจง่าย เวลาเขียนเป็นโปรแกรมแล้วจะสั้นกว่าภาษาอื่น ๆ หลายเท่า ในเรื่องประสิทธิภาพนั้นก็ยังเถียงกันอยู่มาก แต่ส่วนใหญ่ก็ยอมรับว่าทำงานได้ช้า และอาจใช้กันอยู่มาก และอาจใช้กับงานขนาดใหญ่ ๆ (มากกว่า 3,000 เรคอร์ด) ไม่ค่อยได้

โลตัส

โลตัส เป็นภาษาโปรแกรมได้เช่นกัน แต่คำสั่งนั้นจะถูกฝังอยู่ในรูปของคำสั่งให้กระทำต่อข้อมูลในเซลล์ต่าง ๆ การจัดเซลล์นั้นก็ไม่แตกต่างกับการจัดฐานข้อมูลแบบรีเรชันแนลคือแบ่งเป็น "แถว (Row)" และ "คอลัมน์ (Column)" มียูสเซอร์จำนวนไม่น้อยที่สามารถสร้างงานด้วยโปรแกรมสำเร็จรูปอันนี้ นี่แสดงว่าโปรแกรมภาษานี้ง่ายต่อการใช้งาน แต่โปรแกรมนี้อาจถูกแก้ไขโดยผู้ใช้ไม่รู้ตัว หลาย ๆ คนไม่ยอมรับว่านี่เป็นภาษาโปรแกรม แต่ในสหรัฐอเมริกาเคยมีการฟ้องร้องกันในศาล คือมีบริษัท ๆ หนึ่งได้ซื้อโปรแกรมประเภทนี้ไปใช้ แล้วทำคำสั่งวิเคราะห์ทางการเงินไว้ผิด จึงทำให้เกิดการขาดทุนย่อยยับ แล้วไปฟ้องร้องเรียกค่าเสียหายจากผู้ผลิตซอฟต์แวร์ แต่ศาลตัดสินว่าความผิดเป็นของผู้ใช้ ทั้งนี้เพราะไปทำคำสั่ง (Program) ผิดเอง

เมื่อมองดูอย่างกว้าง ๆ เราก็จะบอกว่า ภาษาต่าง ๆ นี้มีข้อแตกต่างกันอยู่มาก และมีผลต่อความยากง่ายในการพัฒนาโปรแกรมพอสมควร แต่เราจะใช้ข้อมูลเหล่านี้เพื่อการตัดสินใจได้ก็เมื่อเราอยู่ในสถานะที่จะเลือกได้ทุกภาษา ซึ่งเรามักจะไม่ได้อยู่ในสถานะดังกล่าว ยกตัวอย่างเช่น ถ้าเราทำงานอยู่ในศูนย์คอมพิวเตอร์ขนาดกลาง และใช้เครื่องเมนเฟรมของยี่ห้ออะไร วันหนึ่งเราอาจไม่มีสิทธิ์เลือกเลย เพราะภาษาที่ใช้กันในเครื่องเมนเฟรมตัวนั้นก็อาจเป็นโคบอลแต่เพียงอย่างเดียว หรือเราเป็นบริษัทขนาดเล็กมาก เราซื้อได้แค่เครื่องไมโครคอมพิวเตอร์ 2-3 ตัว อย่างนี้เราก็ไม่มีโอกาสได้ใช้ภาษาฐานข้อมูลที่ดี ๆ ได้อย่างมากก็ซื้อได้แค่ภาษาดีเบส เราก็ต้องทนใช้กันไป

ภาษาไหนดีกว่ากัน

จากการที่เรามักจะต้องถูกบีบให้ใช้ภาษาใดภาษาหนึ่งอยู่เสมอ ๆ ผมก็อยากจะถามว่า เรายังพอจะทำงานได้หรือไม่ และถ้าสามารถจะเปลี่ยนยี่ห้อหรือเปลี่ยนชนิดของเครื่องไปได้เราจะทำงานได้ดีกว่าเก่าสักเท่าใด เรื่องนี้ผมพบว่าภาษาทุกภาษานั้นสามารถจะทำให้เราสามารถบรรลุเป้าหมายหลักของงาน "ดาต้าโปรเซสซิง" ได้เสมอ และภาษาที่ใช้นั้นไม่ได้ช่วยให้ดีขึ้นเท่าใดนัก ปัญหาส่วนใหญ่จะอยู่ที่การวางโครงสร้าง หรือลอจิก (Logic) ของโปรแกรมมากกว่า บักจะเกิดขึ้นได้จากการคิดผิดมากกว่าการเลือกใช้ภาษาผิด ภาษาบางภาษาอาจมีคำสั่งที่ตรงกับลักษณะงานพอดีอยู่ 2-3 คำสั่ง แต่นั่นจะช่วยเราได้น้อยมาก ถ้าหลักการภายในสมองของเราถูกต้องและแจ่มใส เราก็สามารถสร้างโปรแกรมด้วยคำสั่งที่อ้อม ๆ ได้ และจะทำได้เร็วกว่าด้วยซ้ำไป ยกตัวอย่างเช่น ในภาษาปาสคาลนั้นเขาให้เราตั้งชื่อตัวแปรชนิดบูลีน (Boolean Variable) ได้ด้วย หลาย ๆ คนจะพยายามหาทางใช้ประโยชน์ เช่น อาจตั้งชื่อตัวแปรอันหนึ่งว่า END-SWT และเมื่อโปรแกรมวิ่งไปจนถึงเงื่อนไขอันหนึ่งที่เรารู้ว่าจะต้องจบโปรแกรม เราก็ไปเซตค่า END-SWT ให้เป็น "True" เสร็จแล้วให้โปรแกรมวนลูปกลับไปที่คำสั่ง IF END-SWT THEN GOTO EOJ; โปรแกรมแบบนี้ดูจะหรูขึ้น แต่จริง ๆ แล้วเราไม่จำเป็นต้องประกาศให้ END-SWT เป็นตัวแปรชนิดบูลีนเลย เราอาจให้ END-SWT เป็นตัวเลขธรรมดา และเมื่อโปรแกรมวิ่งไปถึงเงื่อนไขที่ต้องการ เราก็เซ็ตค่า END-SWT เป็นเลข 1 เสร็จแล้วก็เปลี่ยนคำสั่ง IF ให้เป็น IF (END-SWT = 1) THEN GOTO EOJ; ก็ได้ การใช้ตัวเลขธรรมดาเป็นสวิตช์หรือแฟล็ก (Flag) นี้เราใช้กันเป็นประจำอยู่แล้ว เราใช้จนชำนาญอยู่แล้ว ดังนั้นจึงไม่มีความจำเป็นอะไรที่จะต้องเปลี่ยนไปใช้ตัวแปรแบบบูลีน

อนึ่ง ผมอยากจะบอกว่า ภาษาต่าง ๆ ที่เรามีอยู่นี้ ส่วนใหญ่แล้วจะมีคำสั่งให้เลือกใช้จนเกินความจำเป็น และพวกส่วนเกินเหล่านี้มักจะเป็นคำสั่งหรูหรา คือสร้างขึ้นมาเพื่อให้ฟิตกับสถานการณ์พิเศษพิสดารบางอย่าง คนเรา

มักจะไปเสียเวลาคิดหาหนทางที่จะใช้คำสั่งพิเศษพิสดารเหล่านั้น แล้วพลอยทำให้โปรแกรมเกิดเป็นบัก (เพราะเข้าใจคำสั่งนั้นผิด ๆ) ขึ้นมา งานมักจะล่าช้าเพราะโปรแกรมเมอร์มัวแต่ไปพยายามใช้คำสั่งเหล่านี้ ผมเองนั้นชอบแนวคิดแบบ "KISS" ซึ่งย่อมาจากคำว่า Keep It Simple System ผมจะเลือกใช้แต่คำสั่งที่เข้าใจได้ง่าย ๆ เป็นคำสั่งที่ตรงไปตรงมา แนวคิดแบบนี้ไม่ใช่มีแต่ในความฝัน เพราะเรามีคนเขียนเป็นตำรามาเยอะแล้ว เช่น ภาษาฟอร์แทรนนั้น จริง ๆ แล้วใช้แค่ 9 คำสั่งก็เกินพอแล้ว ซึ่งได้แก่ Assignment Statement, If Statement, GO TO Statement,...etc. ในภาษาปาสคาลนั้นผมก็ใช้อยู่เพียง 13-14 คำสั่งเท่านั้น จุดสำคัญก็คือ เมื่อเราวนเวียนใช้อยู่แต่ในกลุ่มย่อยอันนี้ เราจะมีความชำนาญ จะสามารถใช้ได้ถูกต้องและรวดเร็ว เราจะไม่ต้องไปเปิดคู่มืออีกเลย

เมื่อพูดถึงการใช้เพียงคำสั่งที่สั้น ๆ ง่าย ๆ ก็จะมีคนแย้งว่า โปรแกรมแบบนี้จะยาวขึ้น ทั้งนี้เพราะแทนที่จะไปเลือกหาคำสั่งที่ใช้ได้ตรงเป้าเพียงคำสั่งเดียว เราก็อาจต้องใช้คำสั่งง่าย ๆ นั้นหลายคำสั่ง แล้วก็จะมีคำถามว่าประสิทธิภาพของโปรแกรมจะลดลงไปสักเท่าใด เรื่องนี้ผมพบว่าเจ้าคำสั่งที่ซับซ้อนนั้นถึงแม้ว่าจะเขียนเพียงบรรทัดเดียว แต่เมื่อแปลเป็นภาษาเครื่องแล้วมันไม่ใช่ว่าจะสั้นหรือทำได้เร็วเสมอไป นอกจากนี้แล้ว การที่เครื่องจะต้องทำตามคำสั่งมากขึ้นอีกสักเล็กน้อยนั้นส่วนใหญ่จะไม่มีผลกระทบต่อความเร็วในการโต้ตอบ โปรแกรมจะทำงานช้าก็เมื่อมีการสั่งให้อ่านหรือบันทึกข้อมูลมากขึ้น เวลาในการคำนวณนั้นจะน้อยมาก น้อยจนไม่ต้องนำมาใช้ในการประมาณความเร็วก็ได้ โอเวอร์เฮดทางด้านตัวซีพียูนั้นจะเกิดขึ้นในตอนทำอินพุตเอาต์พุตเช็ค (I/O Checking) มากกว่า ยกตัวอย่าง เช่น โปรแกรมที่ทำหน้าที่บริการฝากถอนเงินแบบออนไลน์ของธนาคารพาณิชย์ต่าง ๆ นั้นเขาจะประมาณว่าต้องใช้ซีพียูถึง 200,000 คำสั่งต่อการฝากถอน 1 ครั้ง แต่ขั้นตอนที่เกี่ยวกับการคำนวณและเปรียบเทียบหรือโยกย้ายตัวเลขที่เราเขียนลงไปจะยาวเพียง 1000-2000 คำสั่ง (ระดับภาษาเครื่อง) เท่านั้น ส่วนที่เหลือจะเป็นโอเวอร์เฮดในการทำอินพุตและเอาต์พุต

ทางเลือกการใช้ภาษา

ข้างบนนั้นเป็นข้อแนะนำในกรณีที่เราถูกสภาวะแวดล้อมบังคับอยู่คือเมื่อมีภาษาอะไรให้ใช้เราก็จงรีบใช้ภาษานั้น และเวลาจะใช้ภาษาใดก็ตาม ก็ให้เลือกใช้แต่คำสั่งที่ง่าย ๆ สั้น ๆ คือ พวกที่จะไม่ต้องเปิดดูคู่มือบ่อย ๆ พยายามใช้อยู่แต่ในซับเซตที่เรามีความชำนาญแล้ว แต่อย่างไรก็ตาม งานของเราไม่จำเป็นจะต้องผูกติดกับเครื่อง และยี่ห้อใดยี่ห้อหนึ่งอยู่ตลอดชีวิต ใน 3-5 ปีข้างหน้าเราจะมีโอกาสเปลี่ยนเครื่องหรือเปลี่ยนยี่ห้อ ถึงตอนนั้นเราจะอยู่ในสถานะที่จะเลือกอะไร ๆ ได้มากขึ้น เรื่องนี้เราก็ต้องมาดูแนวโน้มของตลาด ผมแนะนำให้ดูแนวโน้มของตลาดมากกว่าที่จะดู "ความมีประสิทธิภาพ" หรือ "ความเก่ง" ของภาษาโปรแกรม ทั้งนี้เพราะภาษาโปรแกรมที่ว่าดีหรือเก่งนั้นอาจตายได้ ราคาและการให้การสนับสนุนจากบริษัทมีความสำคัญกว่า ภาษาที่บอกว่ายากนั้นอาจกลายเป็นง่าย เมื่อบริษัทมาแนะนำหรือมีตำราดี ๆ ให้อ่าน ราคาที่ถูก (อย่างของบอร์แลนด์อินเตอร์เนชั่นแนล) ก็อาจทำให้ภาษาที่ดูจะยาก (ปาสคาล) กลายเป็นภาษาที่มีคนนิยมได้ แนวโน้มของตลาดในปัจจุบันคือ คนจะหันมาใช้ไมโครคอมพิวเตอร์และพีซีแลนกันมากขึ้น ปริมาณความต้องการโปรแกรมเมอร์ในงานที่ใช้เครื่อง 2 รูปแบบนี้กำลังผ่านจุดกึ่งกลางไปแล้ว ดังนั้นถ้าท่านคิดจะยึดอาชีพเป็นนักคอมพิวเตอร์ต่อไปก็จะต้องหันมาใช้เครื่อง 2 ชนิดนี้ ภาษาที่ใช้ในเครื่อง 2 รูปแบบนี้คือ เบสิก ปาสคาล และซี แต่ภาษาเบสิกนั้นเริ่มลดความนิยมไปแล้ว ปาสคาลกับซีจึงเป็นภาษาสำหรับอนาคต สองภาษานี้ใกล้เคียงกันมาก ภาษาซีอาจดูกว้างกว่ามีคำสั่งให้เลือกมากกว่า แต่ถ้าท่านยึดแนวของ "KISS" ท่านก็ควรจะเน้นหาความชำนาญในภาษาปาสคาลเสียก่อน ท่านคงต้องเริ่มเสียตั้งแต่วันนี้ แต่อย่างไรก็ตาม ท่านจะต้องพร้อมเสมอที่จะกระโดดไปสู่ภาษาซีหรืออื่นใดที่จะมาทดแทน 2 ภาษานี้ในอนาคต

สรุป

โดยสรุปแล้ว ผมเห็นว่าภาษานั้นไม่สำคัญอะไรเลย โดยเฉพาะอย่างยิ่งอย่าไปสนใจในแง่ "ความเก่ง" ของภาษา ภาษาที่เก่ง ๆ นั้นตายไปแล้วเป็นส่วนใหญ่ ภาษาอัลกอร์ (ALGOL) เป็นตัวอย่าง ในสมัยหนึ่งนักวิทยาศาสตร์คอมพิวเตอร์เคยกำหนดว่าตำราทางด้านเทคนิคคอมพิวเตอร์ต้องใช้ภาษานี้เป็นตัวอย่าง หรือพูดง่าย ๆ คือ ยกให้เป็นภาษามาตรฐานในการสื่อความคิด แต่ลงท้ายเราก็ลืมกันจนหมด งานพัฒนาโปรแกรมนั้นต้องการความสามารถในการวางโครงสร้าง (Logic) ของโปรแกรมมากกว่า ถ้าลอจิกดีเสียอย่าง ท่านก็จะล่องลอยไปกับกระแสคลื่นได้โดยไม่มีวันสิ้นสุด